Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Previously Presented) In a system in which a plurality of computers are interconnected over a network, a distributed application control system, wherein

an agent platform owned by each computer comprises:

(a) an agent base, said agent base having:

(a1) a shell agent responsive to an input of a script language controlling the distributed application to interpret and execute said script language;

(a2) a local service agent furnishing information of a local file system to said computer; and

(a3) an application agent controlling the application;

(a4) said agent base furnishing respective fields of execution to said shell agent, local service agent and the application agent;

(b) agent mover furnishing the function of causing an agent to move to an agent base of at least one other computer;

(c) a remote call module furnishing functions for an agent to have communication with an agent of an own computer or at least one other computer; and

(d) agent generator generating an application agent; and

(e) wherein execution of the application distributed over each computer is controlled responsive to an input of said script language.

2. (Original) The distributed application control system as defined in claim 1 wherein said shell agent includes:

a current directory supervising a current directory in executing the script language; and

a shell interpreter having syntax analyzer analyzing the syntax of said script language and command analysis - executor analyzing and executing a command; and wherein

when said current directory indicates an agent base on a remote controller, said shell agent itself is moved by said agent mover to said remote controller to execute an application agent.

3. (Original) The distributed application control system as defined in claim 1 wherein said shell agent further includes a repository path table of an agent; and wherein the application agent is retrieved from the repository path and executed.

4. (Original) The distributed application control system as defined in claim 2 wherein said shell agent further includes a status table for supervising the status of an agent;

said status table has a parallel execution counter for an agent and a terminal execution flag; and wherein

if, when the current base of said current directory is a remote computer, the status of the agent is not the parallel execution nor the terminal execution mode, said shell agent is moved to said remote computer.

5. (Original) The distributed application control system as defined in claim 2 further comprising:

means for invoking a local service agent assisting each computer to access local resources for generating an application agent by said shell agent;

wherein

when said application agent is generated, said shell agent generates the application agent through said local service agent instead of directly invoking said agent generator.

6. (Original) The distributed application control system as defined in claim 2 wherein

said shell agent further includes an agent referencing table; and wherein

a request for movement or command execution is made to an application agent stored in said agent referencing table and booted outside.

7. (Original) The distributed application control system as defined in claim 1 wherein

said application agent includes application booting means dependent on said computer; and wherein

said application is generated and supervised under commands by said shell agent.

8. (Original) The distributed application control system as defined in claim 1 wherein

said application agent includes means for referencing an extension-application associating table to select an application to be booted depending upon the sort of the file extension;

said selected application being generated and supervised under a command of said shell agent.

9. (Original) The distributed application control system as defined in claim 1 wherein

said agent platform further includes thread generator; and wherein

if parallel execution is specified in the input script language, as many threads as are required for executing said script language are generated to control the parallel execution of the application.

10. (Original) The distributed application control system as defined in claim 9 wherein

said agent platform includes a sub-shell agent in an agent base of said remote controller when generating an application agent in said remote computer by the generated thread;

said sub-shell agent taking over the role of generating said application agent.

11. (Previously Presented) A method for controlling the distributed application in a system in which a plurality of computers are interconnected over a network, wherein the method comprises the steps of:

(a) interpreting and executing a script language input to control a distributed application by a shell agent in each agent platform of each computer;

(b) furnishing information pertinent to a local file system to said computer by a local service agent;

(c) controlling the application by an application agent;

(d) furnishing, by an agent base, respective fields of execution for said shell agent, local service agent and said application agent;

(e) allowing said agent to be movable to at least one other agent base of at least one other computer through an agent movement mechanism;

(f) allowing said agent to communicate with an own computer or at least one other computer through a remote call function;

(g) interpreting, by said shell agent, an input script language to control the generation of the application agent through an agent generating function; and

(h) executing the distributed applications in said computers under control responsive to inputting of said script language.

12. (Original) The method for controlling the distributed application as defined in 11 wherein

said shell agent supervises a current directory in executing the script language, said shell agent also analyzing and executing the application distributed across said computers; and wherein

when the current directory indicates an agent base on a remote computer, the shell agent itself is moved to said remote computer to execute the application agent.

13. (Original) The method for controlling the distributed application as defined in claim 11

wherein

said shell agent further references a repository path table of said agent to retrieve the application agent from a repository path table to execute the retrieved application agent.

14. (Original) The method for controlling the distributed application as defined in claim 11 wherein

said shell agent uses information of a terminal execution flag and the parallel execution count of an agent of a status table supervising the agent status and, if a current base of said current directory is a remote computer, the agent status being not a parallel executing status nor a terminal execution mode, the shell agent is moved to the remote computer.

15. (Original) The method for controlling the distributed application as defined in claim 11 wherein, in generating an application agent, the shell agent does not invoke the

agent generating function, with the application agent being generated through a local service agent assisting each computer to access local resources.

16. (Original) The method for controlling the distributed application as defined in claim 11 wherein

said shell agent has an agent referencing table and makes request for movement or command execution to an application agent stored in said agent referencing table and booted outside.

17. (Original) The method for controlling the distributed application as defined in claim 11 wherein

said application agent has a function of computer-dependent booting of an application and generates said application under a command of said shell agent to supervise the so-generated application.

18. (Original) The method for controlling the distributed application as defined in claim 11 wherein

said application agent uses an extension application accommodating table provided in each computer to select an application booted depending on the sort of file extensions, said application agent generating and supervising said generated application under a command of said shell agent.

19. (Original) The method for controlling the distributed application as defined in claim 11 wherein

said agent platform has the function of generating a thread and wherein

when the input script language has a description on the parallel execution, there are generated threads necessary for parallel execution to control the parallel execution of the application.

20. (Original) The method for controlling the distributed application as defined in claim 11 wherein

when said agent platform generates an agent platform in a remote computer by a generated thread, a sub-shell agent is generated in an agent base in the remote computer, said sub-shell agent taking over the role of application agent generation.

21. (Original) In a distributed application control system in which a plurality of computers are interconnected over a network,

a computer readable program product configured for operating a shell agent, local service agent, application agent, agent base, agent mover, remote call module and agent generator on said computers,

said program product comprising commands operating said control system;

said control system comprising:

(a) an agent platform of each computer having a shell agent fed with a script language for controlling distributed applications as input to interpret and execute the script language,

(b) a local service agent providing the computer with information comprising local file system,

(c) an application agent controlling an application,

(d) an agent base furnishing a field of execution to said shell agent, local service agent and the application agent,

(e) agent mover furnishing to an agent a function of agent movement to another agent base of at least one other computer,

(f) a remote call module providing the agent with a function of communicating with an own computer or with another agent of at least one other computer, and

(g) agent generator furnishing a function of generating an application agent, and in which the control system controls the execution of distributed applications in said computers.

22. (Original) In a distributed application control system in which a plurality of computers are interconnected over a network,

a computer accessible medium carrying thereon a program for operating a shell agent, local service agent, application agent, agent base, agent mover, remote call module and agent generator on said computers,

said program comprising commands operating said control system,

said control system comprising:

(a) an agent platform of each computer having a shell agent fed with a script language for controlling distributed applications as input to interpret and execute the script language,

(b) a local service agent providing the computer with information comprising a local file system,

(c) an application agent controlling an application,

(d) an agent base furnishing a field of execution to said shell agent, local service agent and the application agent,

(e) agent mover furnishing to an agent a function of agent movement to another agent base of at least one other computer,

(f) a remote call module providing the agent with a function of communicating with an own computer or with another agent of at least one other computer, and

(g) agent generator furnishing a function of generating an application agent, and in which the control system controls the execution of distributed applications in said computers.

23. (Original) A computer connected to one or more other computers over a network, said computer comprising:

an agent platform, a file system local to said computer, an application and an agent repository;

said agent platform including:

(a) an input unit receiving an input to said computer, an output unit issuing an output from said computer,

(b) a shell agent interpreting and executing an input script language to generate an application agent and to have communication,

(c) a local service agent, as an agent for providing said computer with inherent information or functions, said local service agent supervising a local file system to provide the shell agent with information,

(d) an application agent executing a given task on request from said shell agent;

(e) an agent base furnishing an area for storage of an agent program being executed on each agent;

(f) an agent movement module providing an agent in said agent base with a function of movement to another agent platform;

(g) a thread generating unit furnishing a function of generating a new thread when an agent in an agent base is operated in a multi-thread operation;

(h) a remote call module providing an agent in said agent base with a function of invoking a method from another agent in said agent base or an agent of another agent platform; and

(i) an agent generating module invoked in generating a new agent in said agent base and executing the generated agent, said agent generating module referencing said agent repository to retrieve an agent program to generate a new agent in said agent base based on the retrieved result;

wherein

said script language input through said input unit is interpreted by said shell agent to boot said application agent, said application agent supervising an actual application;

said shell agent and the application agent being movable to at least one other computer through said network, using said agent movement module, said shell agent being able to communicate with an agent in at least one other computer through said network.

24. (Original) The computer as defined in claim 23 wherein

a shell interpreter in said shell agent is fed from said input unit with a script language and interprets the input script language, said shell interpreter executing the parallel processing if the script language is a parallel execution sentence; and wherein

if the script language can be processed as an internal command, said shell interpreter performs control to effect parallel processing, whereas, if otherwise, said shell interpreter performs control to effect external command processing.

25. (Original) The computer as defined in claim 23 in which said shell agent includes a shell interpreter having a syntax analysis unit responsive to an input of a script language from said input unit to interpret the input script language and a command analysis- executing unit for executing a command, a status table for supervising the state of execution of said shell agent, and a current directory storage unit for memorizing the current directory of said shell agent;

wherein

said syntax analysis unit referencing a keyword table having a keyword registered thereon, said syntax analysis unit when receiving an input having a syntax beginning with a keyword of a parallel executing sentence requesting said thread generating unit to generate a new thread to effect parallel execution;

said command analysis- executing unit verifies whether or not an input command is an internal command that can be processed in said shell agent; if, as a result of verification, the input command is the internal command, a command definition is loaded from an internal command definition storage unit, memorizing the internal command definition, to execute the command definition; said agent movement unit being invoked if, in executing a directory movement command from among internal commands loaded from the internal command definition storage unit, movement of said shell agent itself is necessary;

said remote call module being utilized if a method of another agent is executed;

the result of command execution by said command analysis- executing unit being delivered to said output unit and output from said computer;

an agent program for executing an external command being retrieved from an agent repository in an external command execution unit provided in said shell agent in case of execution of an external command; a repository path table being referenced to determine which path in said agent repository is to be retrieved or not to sequentially retrieve the agent repository for paths stored in said repository path table;

wherein

if, as a result of retrieval of said agent repository, the necessary agent is retrieved, said external command execution unit requests the agent generating unit to generate an agent, said agent generating unit so-requested then generating an application agent; and wherein

when making a method call to an agent other than said shell agent, said external command execution unit invokes said remote call module.

26. (Original) The computer as defined in claim 25 wherein said local service agent receives an external command name and the path information stored in said repository path table from said external command execution unit to retrieve said agent repository and to invoke an actual agent generating unit to generate said application agent.

27. (Original) The computer as defined in claim 25 wherein, if said internal command is a command indicating directory movement, the directory of destination of movement is extracted from said internal command to develop the extracted directory of destination of movement in a directory constituent element to store the base name and the directory in a base in a current base of the directory storage table and in an in-base directory, respectively;

the value of a parallel execution counter for storage of depth information of parallel execution, provided in said status table, is checked as to whether or not it is equal to 0; if the parallel execution counter is other than 0, it indicates that the shell agent is executing in parallel; the value of said current directory storage unit is updated to the value of the directory of the destination of movement specified by a command representing said directory movement, with the shell agent not being moved between computers;

if the parallel execution counter is 0, the shell agent is executing in a sole thread, so a terminal mode flag, provided in said status table, and which is set to a first value and to a second value if the shell agent is being executed as a shell awaiting an input from a terminal and if the shell agent is being executed as a batch command without being connected to a specified terminal, respectively, is referenced, with movement between computers of the shell agent by a command representing the directory command not being made, but only the updating of the value of the current directory storage unit being made, when said terminal mode flag is of the first value; and wherein

if said terminal mode flag is of a second value, the batch mode is being executed, so that the value of the current directory storage unit and the directory of destination of movement are compared to each other; if the current directory storage unit and the directory of destination of movement are on the same computer, value of the current directory storage unit is updated; if the current directory storage unit and the directory of the destination of movement are on the same computer, said shell agent is moved in a controlled manner to an agent base of a computer where there exists a current base of the directory of destination of movement.

28. (Original) The computer as defined in claim 23 wherein

in detecting a parallel execution sentence to execute the parallel execution processing in a syntax analysis unit of said shell interpreter,

a parallel execution counter of said status table is incremented by one to record that the shell agent is currently in parallel executing state;

as many threads as there are commands in said parallel executing sentence are generated by said thread generating unit;

the generated number of threads is recorded and set in a thread variable;

the commands in said parallel executing sentence are executed in parallel in respective threads;

on completion of the execution of the respective threads, the number of values of thread variables is decremented by one, with the number of said threads being zero at a time point of end of all threads to terminate parallel execution; and wherein

the value of said parallel execution counter is decremented by one to terminate the execution of the entire parallel execution.

29. (Original) The computer as defined in claim 25

wherein

said external command execution unit checks whether or not the current base stored in said current directory storage unit actually is coincident with the current base actually executing the shell agent; in case of coincidence, said external command execution unit checks whether or not a program of an application agent in question is present in an agent repository path of the agent repository being executed in order to generate an external application agent in the base being executed; if there is such program of the application agent, the external command execution unit asks said agent generating unit to load the application agent program from the agent repository to execute the application agent;

in case of non-coincidence of the current base stored in said current directory storage unit with the base currently executing the shell agent, said remote call module is used to generate a sub-shell agent in said current base; and wherein

it is checked whether or not there is a program of an application agent in question in an agent repository path of the agent repository of the current base in said sub-shell agent; in case such program of said application agent exists, the program of the application agent from

said agent repository is loaded to ask the agent generating unit to load the application agent program from said agent repository to execute the application agent.

30. (Original) The computer as defined in claim 23 wherein

it is checked whether or not the current base stored in said current directory storage unit coincides with the executing base actually executing the shell agent; in case of coincidence, said local service agent of the executing base is asked to execute the external agent to generate the external application agent in the executing base;

said local service agent checks whether or not there is a program of an application agent in question in an agent repository path of an agent repository of a base in execution; if there is any such program of the application agent, said local service agent asks the agent generating unit to load the application agent program from the agent repository to execute the application agent;

in case of non-coincidence between the current base stored in the current directory storage unit with the base in execution actually executing the shell agent, said local service agent uses said remote call module to ask the local service agent to execute the external agent; and wherein

said local service agent checks whether or not there is any program of an application agent in question in an agent repository path of the agent repository of the current base; if there is any such program of the application agent, said local service agent asks the agent generating unit to load the program of the application agent from the agent repository to execute the application agent.

31. (Original) The computer as defined in claim 23

wherein

there is provided a general-purpose application agent as said application agent; an extension application accommodating table memorizing and holding the relation between an extension and the application corresponding to said extension is referenced and retrieved from the extension of a designated file to boot an application having an extension coincident with the extension of the designated file.